Un analyseur LFG efficace pour le français: SxLfg

Pierre Boullier, Benoît Sagot, Lionel Clément

Un analyseur LFG efficace : SXLFG

Pierre Boullier, Benoît Sagot, Lionel Clément
INRIA - Projet Atoll
Domaine de Voluceau, B.P. 105, 78153 Le Chesnay Cedex, France
{pierre.boullier, benoit.sagot}@inria.fr,
lionel.clement@lefff.net

Mots-clefs : syntaxe, analyseur, LFG, désambiguïsation, forêt partagée

Keywords: syntax, parser, LFG, disambiguation, shared forest

Résumé Dans cet article, nous proposons un nouvel analyseur syntaxique, qui repose sur une variante du modèle *Lexical-Functional Grammars* (Grammaires Lexicales Fonctionnelles) ou *LFG*. Cet analyseur LFG accepte en entrée un treillis de mots et calcule ses structures fonctionnelles sur une forêt partagée. Nous présentons également les différentes techniques de rattrapage d'erreurs que nous avons mises en œuvre. Puis nous évaluons cet analyseur sur une grammaire à large couverture du français dans le cadre d'une utilisation à grande échelle sur corpus variés. Nous montrons que cet analyseur est à la fois efficace et robuste.

Abstract In this paper, we introduce a new parser based on the *Lexical-Functional Grammars* formalism (*LFG*). This LFG parser accepts as input word lattices and computes functional structures on a shared forest. We also present various error recovery techniques we have implemented. Afterwards, we evaluate this parser on a large-coverage grammar for French in the framework of a large-scale use on various corpus. We show that our parser is both efficient and robust.

1 Introduction

Pour pallier les difficultés algorithmiques des analyseurs syntaxiques sur du texte tout venant, il est aujourd'hui habituel d'appliquer un mode opératoire robuste (méthodes markoviennes, automates finis, etc.). Ces méthodes sont très satisfaisantes pour un grand nombre d'applications qui ne dépendent pas d'une représentation complexe de la phrase, mais la finesse d'analyse en pâtit tellement qu'il est illusoire d'avoir une représentation du syntagme ou des dépendances non locales qui satisfassent une définition linguistique sérieuse. C'est pour cette raison que nous proposons de bâtir un analyseur syntaxique qui soit à la fois compatible avec une théorie linguistique (ici LFG) et qui soit robuste face à la variabilité des productions langagières.

Le développement d'un nouvel analyseur syntaxique pour le formalisme LFG (*Lexical-Functional Grammars*, cf. p. ex. (Kaplan, 1989)) n'est pas en soi très original. Il en existe déjà un certain nombre, comme ceux de (Kaplan & Maxwell, 1994), (Andrews, 1990), ou (Briffault *et al.*, 1997). Toutefois, ils n'utilisent pas toujours de la manière la plus complète possible les différentes techniques algorithmiques de partage de calcul et de représentation compacte de l'information qui permettent d'écrire un analyseur efficace bien que le formalisme LFG, comme de nombreux formalismes qui reposent sur l'unification, soit NP-complet.

Pour utiliser au maximum ces techniques, il nous a donc fallu adapter LFG sans pour autant diminuer son pouvoir d'expression. Associée à un analyseur non-contextuel (CF) tabulaire, cette variante de LFG nous permet d'effectuer efficacement l'analyse d'énoncés complexes. La construction des structures de constituance ne pose théoriquement[1] pas de problème particulier, car elles sont décrites par des grammaires non-contextuelles (CFG) sous-jacentes aux LFG et appelées ici grammaires *support*. Mais la construction efficace des structures fonctionnelles est beaucoup plus problématique. Nous avons développé un module de calcul de ces structures qui partage les sous-structures communes à plusieurs analyses. De plus, des mécanismes de rattrapage d'erreur à tous les niveaux en font un analyseur robuste. Cet analyseur, appelé SXLFG, est décrit ci-dessous puis évalué avec une grammaire du français sur des corpus variés.

2 L'analyseur SXLFG : analyse standard

2.1 L'analyseur Earley

Le moteur de SXLFG est un analyseur CF général qui traite la grammaire support de la LFG. L'ensemble des analyses qu'il produit est représenté sous la forme d'une forêt partagée[2]. L'évaluation fonctionnelle se fait dans une seconde phase au cours d'un parcours bas-haut de cette forêt[3]. L'entrée de l'analyseur est un treillis de mots transformé par le *lexeur* en un treillis de lexèmes (terminaux de la CFG et structures fonctionnelles sous-spécifiées associées). Un post-traitement (facultatif) permet alors de désambiguïser.

L'analyse de la grammaire support est réalisée par une évolution de l'analyseur Earley décrit dans (Boullier, 2003) : il prend en entrée des treillis de mots et permet de récupérer les erreurs syntaxiques (cf. section 3.1). Traiter un treillis en entrée ne nécessite pas, d'un point de vue théorique, des changements considérables à l'algorithme Earley, même aidé d'un guide régulier.

[1] Même si, en pratique, la disponibilité d'un *bon* analyseur est déjà plus délicate.

[2] Rappelons que cette structure permet de représenter en une taille polynomiale en n, nombre de mots du source, l'ensemble potentiellement non borné des arbres d'analyse.

[3] Ce parcours assure que si un symbole se trouve en partie droite d'une production reconnue, toutes les structures fonctionnelles associées à ce symbole ont déjà été calculées (nos forêts partagées sont non cycliques).

2.2 Calcul des structures fonctionnelles

Disposant d'une forêt partagée en sortie de l'analyse CFG, nous devons maintenant calculer les structures fonctionnelles. Bien entendu, la méthode qui consiste à *déplier* la forêt pour en extraire chaque arbre sur lequel on évalue les structures fonctionnelles est impratiquable en termes de temps de calcul. En revanche, l'autre possibilité, une évaluation des structures fonctionnelles directement sur la forêt partagée, est toujours un sujet de recherche. Le problème se simplifie cependant si l'on suppose, comme c'est le cas dans SXLFG, que l'évaluation des équations fonctionnelles associées à une production CFG ne modifie pas les structures fonctionnelles associées aux symboles de sa partie droite. Cette légère restriction dans l'écriture des équations fonctionnelles ne diminue pas pour autant le pouvoir d'expression.

La conséquence directe de cette évaluation *bottom-up* des structures fonctionnelles est que toute sous-forêt n'est évaluée qu'une seule fois et son calcul partagé entre tous ses parents. L'autre conséquence est qu'à chaque nœud de la forêt est associée non pas une structure fonctionnelle unique mais une disjonction de structures fonctionnelles. Très souvent, le résultat de cette évaluation est donc un grand nombre de structures fonctionnelles associées à la racine de la forêt.

2.3 Désambiguïsation

La sortie de l'étape précédente (sauf échec, voir partie suivante) est une forêt partagée de structures de constituants associée à un ensemble de structures fonctionnelles avec partage de structures communes. Ces informations peuvent être la description d'une ou de plusieurs analyses. Il faut donc pouvoir désambiguïser, c'est-à-dire choisir parmi ces analyses celle qui est la plus vraisemblable. Deux familles de techniques sont envisageables : les techniques probabilistes et les techniques à règles. Suivant sur ce point (Clément & Kinyon, 2001), nous utilisons un ensemble de règles qui est une refonte et une extension des trois principes simples qu'ils énoncent et qui s'applique sur les structures fonctionnelles[4]. Chacune de nos règles est appliquée successivement (on peut en changer l'ordre, voire ne pas toutes les appliquer). L'application d'une règle consiste à éliminer les analyses qui ne sont pas optimales au sens de cette règle[5].

À l'issue de ce mécanisme de désambiguïsation sur les structures fonctionnelles, la forêt d'analyse (qui représente les structures en constituants) est filtrée afin qu'elle corresponde exactement aux structures fonctionnelles retenues. En particulier, si la désambiguïsation est complète, ce filtrage rend en général une structure en constituants unique (un arbre).

[4] Cf. (Kinyon, 2000) pour une argumentation sur l'importance de désambiguïsation en se fondant sur des structures comme les arbres de dérivation TAG ou les structures fonctionnelles LFG et non sur celles en constituants.

[5] Nos règles, dans leur ordre d'application par défaut, sont :

règle 1 : *Préférer les analyses maximisant la somme des poids des lexèmes utilisés* ; parmi les entrées lexicales de poids supérieur à la moyenne se trouvent les multi-mots, qui sont ainsi favorisés.

règle 2 : *Préférer les noms communs avec déterminant.*

règle 3 : *Préférer les arguments aux modifieurs, et les relations auxiliaire-participe aux arguments* (le calcul se fait récursivement sur toutes les (sous-)structures).

règle 4 : *Préférer les arguments les plus proches* (même remarque).

règle 5 : *Préférer les structures les plus enchâssées.*

règle 6 : *Trier les structures selon le mode des verbes* (on préfère récursivement les structures à l'indicatif à celles au subjonctif, et ainsi de suite).

règle 7 : *Trier selon les catégories des gouverneurs d'adverbes.*

règle 8 : *Choisir une analyse au hasard* (pour garantir qu'on rende une analyse et une seule).

3 Mécanismes pour l'analyse robuste

3.1 Rattrapage d'erreur pendant l'analyse

La détection d'une erreur dans l'analyseur Earley peut être la manifestation de deux phénomènes : la CFG support n'est pas assez couvrante ou l'énoncé n'est pas du français. Bien entendu, même si l'analyseur ne distingue pas ces deux situations, le concepteur de la grammaire doit y réagir différemment. Le traitement des erreurs dans les analyseurs est un sujet de recherche qui a surtout été abordé dans le cas déterministe et très peu dans le cas des analyseurs CF généraux. Pour des raisons de place, nous ne pouvons décrire ici le mécanisme général de rattrapage CFG que nous avons développé. Il fera l'objet d'une publication ultérieure.

Le calcul des structures fonctionnelles échoue si et seulement si aucune structure fonctionnelle n'est associée à la racine de la forêt partagée. Cette situation d'erreur provient du fait que les contraintes (d'unification) spécifiées par les équations fonctionnelles n'ont pas pu toutes être vérifiées ou que les structures fonctionnelles résultantes sont incohérentes.

Un premier échec déclenche une deuxième évaluation des structures fonctionnelles sur la forêt partagée, au cours de laquelle les vérifications de cohérence sont supprimées. En cas de succès, on obtient à la racine un certain nombre de structures fonctionnelles incohérentes. Si cette seconde tentative échoue, on recherche dans la forêt partagée tous les nœuds *maximaux* qui ont des structures fonctionnelles et dont aucun des pères n'a de structure fonctionnelle. Ils correspondent donc à des analyses partielles disjointes éventuellement incohérentes[6].

3.2 Sur-segmentation des énoncés inanalysables

Malgré les mécanismes exposés précédemment, il arrive que l'analyseur SXLFG ne rende aucune analyse. Ceci peut être dû à l'expiration d'un délai maximum que l'on peut donner en paramètre (*time-out*), ou au fait que le rattrapage d'erreur de l'analyseur Earley n'a pas été capable de produire une analyse raisonnable. La cause peut en être l'insuffisance de la couverture de la grammaire ou un énoncé d'entrée par trop déraisonnable.

Pour cette raison, nous avons réalisé une surcouche à SXLFG qui permet une *sur-segmentation* des énoncés agrammaticaux. L'idée est qu'il arrive fréquemment que des portions de l'énoncé d'entrée soient analysables en tant que phrases, alors même que l'énoncé d'entrée dans son ensemble ne l'est pas. Nous découpons donc en *segments* les énoncés inanalysables (découpage de niveau 1), puis, le cas échéant, redécoupons en segments les segments de niveau 1 inanalysables[7] (découpage de niveau 2) et ainsi de suite. Les niveaux de découpage correspondent successivement aux frontières probables de phrases, aux ponctuations fortes, aux ponctuations faibles, aux coordonnants, et enfin aux frontières de mots.

La qualité de l'analyse décroît évidemment avec le niveau de découpage. Si le découpage de niveau 1 ne pose aucun problème, des difficultés apparaissent au niveau 2. Les niveaux 3 et 4 sont véritablement du rattrapage. Et le niveau 5 n'est là que pour analyser toutes les phrases possibles, et en particulier celles dont on sait analyser certains morceaux de niveau 1 ou 2.

[6] Le processus de désambiguïsation présenté à la section 2.3 s'applique alors à tous les nœuds maximaux.

[7] Un énoncé peut être découpé en deux segments de niveau 1 dont le premier est analysable. Seul le second sera alors sur-segmenté au niveau 2. Et seuls les segments de niveau 2 inanalysables seront sur-segmentés, etc.

4 Mise en œuvre et évaluation

4.1 Mise en œuvre

Nous avons utilisé SXLFG à grande échelle pendant la campagne EASy d'évaluation des analyseurs syntaxiques. Nous l'avons couplé avec une grammaire LFG développée pour XLFG (Clément & Kinyon, 2001), étendue et adaptée aux contraintes liées à ce que SXLFG calcule de manière *bottom-up* les structures fonctionnelles sur la forêt d'analyse CFG. Le lexique et la chaîne de traitement pré-syntaxique mis en œuvre sont décrits dans (Boullier *et al.*, 2005).

4.2 Évaluation

Dans cette section, nous n'évaluerons pas la *qualité* d'une analyse qui dépend pour l'essentiel de la grammaire et qui nécessiterait de disposer d'un corpus de référence annoté manuellement[8]. Nous nous concentrons ici sur l'efficacité de notre système en présentant les résultats obtenus pendant la campagne EASy et sur les corpus EUROTRA et TSNLP.

Corpus	#phrases	couverture (sans vérif. de coh.)	couverture (avec vérif. de coh.)	temps d'analyse			
				moy.	méd.	$\geq$ 0.1s	$\geq$ 1s
EUROTRA	334	94.61%	84.43%	0.33s	0.02s	22.2%	6.0%
TSNLP	1661	98.50%	79.12%	0.03s	0.00s	2.8%	0.6%
EASy	40859	66.62%	41.95%	n.d.[10]			

TAB. 1 – Évaluation de SXLFG, avec un *time-out* de 15 secondes[9].

		Corpus complet	Phrases valides pour la CFG support	
		Analyse CFG	Analyse CFG	Analyse complète
Données sur les corpus[11]	#phrases	40859	35756	
	n_{moy} / n_{max}	20.95 / 541	19.06 / 173	
	UW_{moy} / UW_{max}	0.79 / 97	0.75 / 65	
Temps d'analyse	moy	0.05s	0.01s	3.35s
	med	0.00s	0.00s	0.03s
	$<$ 0.1s	98.2%	98.8%	57.8%
	$<$ 1s	99.8%	99.9%	71.0%
Nombre d'analyses	max	3.10^{73}	5.10^{52}	1^{12}
	med	32 028	29 582	1
	$\geq 10^6$	36.13%	35.28%	0%
	$\geq 10^{12}$	8.86%	7.84%	0%

TAB. 2 – Données sur le EASy corpus, les temps et les nombres d'analyses, avant application de l'heuristique de sur-segmentation (*time-out* de 15 secondes[9]).

[8] Cette qualité dépend aussi des heuristiques de désambiguïsation utilisées et du traitement de la robustesse.

[9] Un *time-out* plus élevé aurait augmenté les taux de couverture mais également les temps d'analyse.

[10] Nous n'avons pas conservé les informations permettant de donner les résultats sur l'ensemble du corpus. Toutefois, la table 2 donne les temps d'analyse pour les 87.51% de phrases reconnues par la CFG support.

[11] n désigne un nombre de mots, et UW un nombre de mots inconnus.

[12] Dans 14.34% des cas, aucune analyse n'a été trouvée en moins de 15s. C'est dans ces cas-là que sont alors appliquées les heuristiques de sur-segmentation.

5 Conclusion

Dans cet article, nous avons introduit l'analyseur SXLFG. À notre connaissance, c'est la première fois qu'un système d'analyse fondé sur le modèle LFG traite du texte tout venant de façon efficace et robuste sans que le pouvoir expressif du formalisme ne soit dégradé. Il est en outre possible de décrire des phénomènes complexes dans SXLFG en accord avec les nombreux travaux linguistiques qui s'y rapportent.

Les expériences relatées utilisent une grammaire du français et un lexique morpho-syntaxique que nous avons également réalisés. Les résultats extrêmement encourageants obtenus ne doivent bien entendu pas masquer qu'il s'agit d'une première tentative qui doit se poursuivre et qui peut être améliorée. Les perfectionnements possibles concernent le formalisme, la grammaire du français et l'analyseur SXLFG lui-même.

Nous avons quelques idées pour étendre notre variante de LFG qui pourraient faciliter certains traitements, en particulier celui des coordonnées. La grammaire doit être étendue et améliorée. En effet, certaines constructions comme les clivées, les comparatives, les coordinations à ellipse, et d'autres, ne sont pas couvertes. D'autre part, la grammaire support (CFG) doit être affinée car son ambiguïté actuelle est déraisonnable (voir section 4.2). Même si notre analyseur Earley y est relativement peu sensible, elle peut rendre prohibitif le temps d'évaluation des structures fonctionnelles associées. Les autres pistes de recherche sur l'analyseur proprement dit concernent essentiellement l'amélioration de la robustesse et du temps de calcul des structures fonctionnelles.

Références

ANDREWS A. (1990). *Functional closure in LFG*. Rapport interne, The Australian National University.

BOULLIER P. (2003). Guided Earley parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT'03)*, p. 43–54, Nancy, France.

BOULLIER P., CLÉMENT L., SAGOT B. & ÉRIC VILLEMONTE DE LA CLERGERIE (2005). Chaînes de traitement syntaxique. In *Actes de TALN 05*, Dourdan, France.

BRIFFAULT X., CHIBOUT K., SABAH G. & VAPILLON J. (1997). An object-oriented linguistic engineering environment using LFG (Lexical-Functional Grammar) and CG (Conceptual Graphs). In *Proceedings of Computational Environments for Grammar Development and Linguistic Engineering, ACL'97 Workshop*.

CLÉMENT L. & KINYON A. (2001). XLFG – an LFG parsing scheme for French. In *Proceedings of LFG'01*, Hong Kong.

KAPLAN R. (1989). The formal architecture of lexical functionnal grammar. *Journal of Informations Science and Engineering*.

KAPLAN R. M. & MAXWELL J. T. (1994). *Grammar Writer's Workbench, Version 2.0*. Rapport interne, Xerox Corporation.

KINYON A. (2000). Are structural principles useful for automatic disambiguation ? In *Proceedings of in COGSCI'00*, Philadelphia, Pennsylvania, United States.